AD-A058 455 AEROSPACE CORP EL SEGUNDO CALIF F/G 12/1

A COMPACT ALGORITHM FOR COMPUTING THE STATIONARY POINT OF A QUA--ETC(U)

JUL 78 J T BETTS F04701-77-C-0078

UNCLASSIFIED TR-0078(3901-03)-1 SAMSO-TR-78-44 NL

1 OF 1

AD
A058455

END

DATE
FILMED

11-78

DDC

REPORT SAMSO-TR-78-44

LEVEL II

2
NW

A Compact Algorithm for Computing the Stationary Point of a Quadratic Function Subject to Linear Constraints

J. T. BETTS

~~Engineering~~ Group
The Aerospace Corporation
El Segundo, Calif. 90245

8 July 1978

Prepared for
SPACE AND MISSILE SYSTEMS ORGANIZATION
AIR FORCE SYSTEMS COMMAND
Los Angeles Air Force Station
P.O. Box 92960, Worldway Postal Center
Los Angeles, Calif. 90009

This final report was submitted by The Aerospace Corporation, El Segundo, CA 90245, under Contract F04701-77-C-0078 with the Space and Missile Systems Organization, Deputy for Advanced Space Programs, P.O. Box 92960, Worldway Postal Center, Los Angeles, CA 90009. It was reviewed and approved for The Aerospace Corporation by J. R. Allder, Engineering Group. First Lieutenant A. G. Fernandez, YCPT, was the Deputy for Advanced Space Programs project engineer.

This report has been reviewed by the Information Office (OI) and is releasable to the National Technical Information Service (NTIS). At NTIS, it will be available to the general public, including foreign nations.

This technical report has been reviewed and approved for publication. Publication of this report does not constitute Air Force approval of the report's findings or conclusions. It is published only for the exchange and stimulation of ideas.

A. G. Fernandez, 1st Lt, USAF
Project Engineer

Robert W. Lindemuth
Robert W. Lindemuth, Lt Col, USAF
Chief, Technology Plans Division

FOR THE COMMANDER

Leonard E. Baltzell
LEONARD E. BALTZELL, Col, USAF, Asst.
Deputy for Advanced Space Programs

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER: SAMSO TR-78-44	2. GOVT ACCESSION NO.
	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle): A COMPACT ALGORITHM FOR COMPUTING THE STATIONARY POINT OF A QUADRATIC FUNCTION SUBJECT TO LINEAR CONSTRAINTS.	5. TYPE OF REPORT & PERIOD COVERED: Final Report.
	6. PERFORMING ORG. REPORT NUMBER: TR-0078(3901-03)-1
7. AUTHOR(s): J. T. Betts	8. CONTRACT OR GRANT NUMBER(s): F04701-77-C-0078
9. PERFORMING ORGANIZATION NAME AND ADDRESS: The Aerospace Corporation, El Segundo, California 90245	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS: Space and Missile Systems Organization, P.O. Box 92960, Worldway Postal Center, Los Angeles, California 90009	12. REPORT DATE: 8 July 1978
	13. NUMBER OF PAGES: 26
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office): 21p.	15. SECURITY CLASS. (of this report): Unclassified
	15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Quadratic Programming
Nonlinear Programming
Orthogonal Decomposition
Householder Decomposition
Constrained Optimization

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

An algorithm is presented for computing the stationary point of a quadratic function of n variables subject to a set of $m (m \leq n)$ linear equality constraints. The procedure is compact in the sense that it requires no two-dimensional arrays of computer storage beyond that needed to store the problem data. The use of a Householder orthogonal decomposition by the method should not degrade the numerical conditioning of the original problem. The method is applicable to problems with singular Hessian matrices, and can be adapted for use in a general quadratic programming algorithm.

DD FORM 1473 (FACSIMILE)

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

009 500 LB

78 08 31 023

AEROSPACE FORM 4341-1 REV 5-74

CONTENTS

1. INTRODUCTION 3

2. PRELIMINARIES. 5

3. ORTHOGONAL DECOMPOSITION ALGORITHM 9

4. COMPUTATIONAL ALGORITHM 13

5. SUMMARY 17

APPENDIX. THE STATIONARY POINT OF A QUADRATIC FUNCTION SUBJECT TO LINEAR CONSTRAINTS. . . 19

1. INTRODUCTION

This report presents an algorithm for computing the stationary point of a quadratic function of n variables subject to a set of $m (m \leq n)$ linear equality constraints. The procedure is compact in the sense that it requires no two-dimensional arrays of computer storage beyond that needed to store the problem data. The use of a Householder orthogonal decomposition by this method should not degrade the numerical conditioning of the original problem. This method is applicable to problems with singular Hessian matrices, and can be adapted for use in a general quadratic programming algorithm.

In the subsequent sections of this report, the identifying numbers of equations in the text are enclosed with parentheses, and the identifying numbers of references are enclosed with brackets.

2. PRELIMINARIES

Define the quadratic function

$$f(x) = \tfrac{1}{2} x^T A x + b^T x \tag{1}$$

and the linear constraints

$$Cx = d \tag{2}$$

where A is an n x n symmetric matrix, b is an n-vector, C is an m x n matrix of rank m, d is an m-vector, and x is an n-vector for $m \leq n$. Define the solution x^* to be the vector which: (a) satisfies the constraints, (b) minimizes the norm of the gradient of f restricted to the constraint surface and, (c) minimizes the length of the orthogonal projection of x on the constraint surface.

When A is positive (negative) definite, the solution defines the unique stationary point which corresponds to the minimum (maximum) of f restricted to the constraint surface. While the stated problem may be of interest by itself, typically it may appear as a subproblem in a more general application. For example, many quadratic programming algorithms solve a series of problems of this type with different constraint sets. Furthermore, an algorithm designed to optimize a non-quadratic function subject to nonlinear constraints may pose a series of quadratic-linear problems to approximate the behavior of the actual functions. Consequently, it is desirable to develop a computational algorithm which will compute a solution to the problem without restricting the rank of A. The computed solution should be the unique solution when it exists, and should be uniquely defined by the algorithm when a unique solution does not exist.

When the minimum norm of the projected gradient is zero, the solution to the stated problem is a stationary point of the Lagrangian function

$$L(x, \lambda) = \tfrac{1}{2} x^T A x + b^T x + \lambda^T (Cx - d) \qquad (3)$$

where λ is the m-vector of Lagrange multipliers. Setting the derivative of this function with respect to x and λ equal to zero yields the set of linear equations

$$\begin{bmatrix} A & C^T \\ C & O \end{bmatrix} \begin{bmatrix} x^* \\ \lambda^* \end{bmatrix} = \begin{bmatrix} -b \\ d \end{bmatrix} . \qquad (4)$$

The optimal solution x^* and the corresponding multiplier values λ^* can be obtained by solving the system (4). It is not necessary to assume that A is of full rank. When the problem has a unique solution, the system may be solved using a suitable algorithm for linear equations. If the possibility of a non-unique solution exists, the system may be solved as a linear least squares problem. This approach has been used[1,2] utilizing the linear least squares algorithm of Hanson and Lawson[3]. A defect in this approach is the need to store the (n + m) x (n + m) coefficient array.

An approach which can be implemented using only the storage required for A and C can be derived by inverting the coefficient matrix in a partitioned

---

[1] J. T. Betts, "A Gradient Projection - Multiplier Method for Nonlinear Programming," Journal of Optimization Theory and Applications, Vol 24, No. 4 (April 1978).

[2] J. T. Betts, "An Accelerated Multiplier Method for Nonlinear Programming," Journal of Optimization Theory and Applications, Vol 21, No. 2 (February 1977).

[3] R. J. Hanson and C. L. Lawson, "Solving Least Squares Problems," Prentice-Hall, Englewood Cliffs, N.J. (1974).

form. It is easily demonstrated that

$$\begin{bmatrix} A & C^T \\ C & O \end{bmatrix}^{-1} = \begin{bmatrix} B_1 & B_2 \\ B_2^T & B_3 \end{bmatrix} \quad (5)$$

where

$$B_1 = A^{-1} - A^{-1} C^T M^{-1} CA^{-1} \quad (6)$$

$$B_2 = A^{-1} C^T M^{-1} \quad (7)$$

$$B_3 = M^{-1} \quad (8)$$

and

$$M = CA^{-1} C^T \quad (9)$$

The partitioned form of the inverse plays an important role in a number of quadratic programming algorithms, including those of Goldfarb[4] and Fletcher[5], as well as in the constrained minimization algorithm of Murtagh and Sargent[6]. Two significant points deserve comment regarding this approach. First, if it is assumed that A^{-1} exists, the submatrices in (5) can be computed directly. Goldfarb, Murtagh, and Sargent assume that A is

---

[4] D. Goldfarb, "Extension of Newton's Method and Simplex Methods for Solving Quadratic Programs," In Numerical Methods for Nonlinear Optimization, F. A. Lootsman (Ed.) Academic Press, London, ch. 17 (1972).

[5] R. Fletcher, "A General Quadratic Programming Algorithm," J. Inst. Math. Appl., Vol 8 (1971).

[6] B. A. Murtagh and R. W. H. Sargent, "A Constrained Minimization Method for Quadratic Convergence," In Optimization, R. Fletcher (Ed.), Academic Press, London, ch. 14 (1969)

positive definite, which ensures that A is of full rank. If the algorithm is part of a more general nonlinear programming algorithm as in [1] and [2], it is restrictive to assume that A is of full rank. For example, the approach could not be used as part of an algorithm to optimize a linear function subject to nonlinear constraints. In these general applications, it is really only necessary that the Hessian matrix restricted to the constraint surface be definite. Fletcher does not assume A is definite, noting that the partitions B_1, B_2, and B_3 must exist if the solution to the problem is unique. However, to compute the initial submatrices in the computer implementation of his quadratic programming algorithm[7], it is necessary to invert the full $(n + m) \times (n + m)$ matrix.

Even if A is assumed to be definite, the partitioned approach to the problem suffers from a second defect. This occurs when $A = I$, $M = CC^T$, which is referred to as the normal matrix. In this case, the condition number of M is the square of the condition number of C and it is generally recognized that the formation of M is to be avoided.

In summary, direct solution of the system (4) is not compact from a storage standpoint. The various forms of solving the partitioned system (5), although compact, require operations which can degrade the numerical conditioning of the given problem and are arbitrarily restrictive with regard to Hessian matrix A. A new algorithm will be proposed which is compact, does not degrade the numerical conditioning, and makes no restrictions concerning the rank of A.

[7] R. Fletcher, "A FORTRAN Subroutine for Quadratic Programming," Report No. AERE R6370, UKAEA Research Group, Harwell, England.

3. ORTHOGONAL DECOMPOSITION ALGORITHM

In this section, an algorithm is developed for solving the stated constrained stationary point problem using an orthogonal decomposition of the constraint matrix. The algorithm is an extension of the linearly constrained linear least squares algorithm LSE given in [3], and makes use of Theorem (3.19) and Theorem (2.3) stated therein.

Define the orthogonal decomposition of C by

$$C = RK^T \tag{10}$$

where K is an n x n orthogonal matrix and R

$$R = [R_{11}\ O] \tag{11}$$

where R_{11} is an m x m nonsingular triangular matrix. Substituting (10) into (2)

$$RK^T x = d. \tag{12}$$

Define the n-vector y by

$$y = K^T x \tag{13}$$

and the partitions of K and y

$$K = [\underbrace{K_1}_{m}\ \underbrace{K_2}_{n-m}] \tag{14}$$

$$y = \begin{bmatrix} y_1 \\ \\ y_2 \end{bmatrix} \begin{matrix} \}m \\ \\ \}n-m \end{matrix} \tag{15}$$

From (11), (13), and (15), one can write (12) as

$$R\,K^T x = Ry = [R_{11}\ O]\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = R_{11} y_1 = d \tag{16}$$

Since R_{11} is non-singular, (16) determines the m-vector y_1. Call the solution $\hat{y}_1$. The (n-m) -vector y_2 is arbitrary.

Pre-multiply (13) by K to give

$$Ky = KK^T x = x \tag{17}$$

where $KK^T = I$, since K is orthogonal. From (14) and (15), (17) becomes

$$x = Ky = [K_1\,K_2]\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = K_1 y_1 + K_2 y_2 \tag{18}$$

Observe that all points satisfying the constraints in Eq. (2) can be represented as functions of the n-m parameters y_2 when the solution of (16) $\hat{y}_1$ is substituted in Eq. (18). In fact if there were no other conditions to satisfy, a reasonable choice for the arbitrary parameters y_2 would be zero, in which case x is the minimum norm solution to the constraints.

Instead of setting $y_2 = 0$, the choice of y_2 shall be determined by a different criterion. Substitute the parametric representation of x from (18) with $y_1 = \hat{y}_1$ into (1) to obtain

$$\begin{aligned} f = {} & \tfrac{1}{2}(K_1\hat{y}_1 + K_2 y_2)^T \ A\,(K_1\hat{y} + K_2 y_2) \\ & + b^T (K_1\hat{y}_1 + K_2 y_2)\,. \end{aligned} \tag{19}$$

The gradient with respect to the variables y_2 is

$$\nabla f = K_2{}^T A\,(K_1\hat{y}_1 + K_2 y_2) + K_2{}^T b. \tag{20}$$

Let us define $\hat{y}_2$ to be the value of y_2 which minimizes

$$\|\nabla f\| = \|K_2^T A K_2 y_2 + (K_2^T b + K_2^T A K_1 \hat{y}_1)\| \tag{21}$$

and is of minimum norm, i.e., minimizes $\|y_2\|$. If a stationary point of f restricted to the constraint surface exists, then $\|\nabla f\| = 0$ and $\hat{y}_2$ defines the optimal value. If the matrix $K_2^T A K_2$ is indefinite, the minimum norm criterion uniquely determines $\hat{y}_2$. In fact, when the rank of $K_2^T AK_2$ is zero as is the case for a linear objective function, the solution which minimizes the norm of $\|y_2\|$ is just $\hat{y}_2 = 0$. Notice also that the formation of the matrix $K_2^T A K_2$ should not degrade the conditioning of the original problem. Observe also that a solution which minimizes $\|y_2\|$ minimizes $\|K_2 y_2\|$ since $\|y_2\| = \|K_2 y_2\|$, and $K_2 y_2$ is just the orthogonal component of x in the constraint surface.

In summary, the original constrained optimization problem is replaced by a lower dimensional unconstrained least squares problem in the variables y_2, after choosing the variables y_1 to satisfy the constraints. The method has the property that the unique minimum length solution of the derived unconstrained problem defines the unique solution of the constrained problem when it exists. When the constrained problem has no unique solution, the algorithm computes a unique point which satisfies the constraints, minimizes the norm of the gradient on the constraint surface, and minimizes the length of the orthogonal component in the constraint surface.

4. COMPUTATIONAL ALGORITHM

In this section a computational procedure based on the approach derived in Section 3 is developed. The procedure is organized so that no additional two-dimensional arrays are needed. Specifically, the original problem data stored in A, C and b is destroyed by the algorithm. Quantities written with a tilde can replace quantities without a tilde in storage, and quantities written with a circumflex can overwrite quantities written with a tilde.

Step 1. Compute the orthogonal matrix K and postmultiply C by it to triangularize C, i.e.,

$$CK = [\underbrace{\tilde{C}_1}_{m} \quad \underbrace{O}_{n-m}] \Big\} m \tag{22}$$

Step 2. Compute

$$\tilde{A} = K^T A \tag{23}$$

Step 3. Form the last n-m rows of the matrix $\hat{A}$ where

$$\hat{A} = \tilde{A} K \tag{24}$$

Observe that from (23) and (24)

$$\hat{A} = K^T AK = \begin{array}{c} \begin{bmatrix} K_1{}^T A K_1 & K_1{}^T A K_2 \\ K_2{}^T A K_1 & K_2{}^T A K_2 \end{bmatrix} \begin{array}{l} \} m \\ \} n-m \end{array} \\ \begin{array}{cc} \underline{\;m\;} & \underline{\;n-m\;} \end{array} \end{array} \tag{25}$$

$$= \begin{bmatrix} \hat{A}_{11} & \hat{A}_{12} \\ \hat{A}_{21} & \hat{A}_{22} \end{bmatrix}$$

Step 4. Compute

$$\tilde{b} = K^T b \tag{26}$$

Step 5. Solve the lower triangular system

$$\tilde{C}_1 y_1 = d \tag{27}$$

for the m-vector $\hat{y}_1$.

Step 6. Compute

$$\hat{b}_2 = -\tilde{b}_2 - \hat{A}_{21}\hat{y}_1 \tag{28}$$

where

$$\hat{b} = \begin{bmatrix} \hat{b}_1 \\ \hat{b}_2 \end{bmatrix} \begin{matrix} \}m \\ \}n-m \end{matrix} \qquad \tilde{b} = \begin{bmatrix} \tilde{b}_1 \\ \tilde{b}_2 \end{bmatrix} \begin{matrix} \}m \\ \}n-m \end{matrix} \tag{29}$$

Step 7. Determine $\hat{y}_2$ as the minimum length solution of the linear least square problem

$$\min \| \hat{A}_{22}\, y_2 - \hat{b}_2 \| . \tag{30}$$

Observe that this process is equivalent to solving (21)

Step 8. Construct the solution vector

$$x = K\,\hat{y} \tag{31}$$

using $\hat{y}_1$ as computed in Step 5, and $\hat{y}_2$ from Step 7.

The algorithm described has been implemented in the subroutine HSQP. The FORTRAN listing of this subroutine is found in [8]. The subroutine makes extensive use of the subroutines HFTI and H12 which implement the algorithms referred to as HFTI, H1, and H2 in [3]. The subroutine HFTI computes the minimum length solution to a linear least squares problem. HFTI requires storage for the problem data and three one-dimensional work arrays. The subroutine H12 implements algorithm H1 and H2 for the construction and application of a Householder transformation. Using H12 it is not necessary to explicitly form the orthogonal matrix K of Eq. (10). Instead, the elements necessary to construct the matrix can be stored in the upper triangular portion of the original matrix C and some one-dimensional work arrays. Successive applications of the matrix K to other vectors implicitly reconstruct the original transformations.

The total storage required for subroutine HSQP, including that required to specify the problem data, is $N_1 = n(m + n) + 5n - m + 4$. In contrast, any algorithm which solves (4) directly will require at least $N_2 = (n + m)(n + m + 1)$ storage locations. Consequently, for some problems N_2 can be nearly twice as large as N_1. The algorithm is used repeatedly as part of the general nonlinear programming algorithm described in [1]. In particular, all of the extrapolation steps used in the constraint phase of this algorithm employ HSQP. Computational experience with the algorithm includes its use to solve the set of 17 equality constrained and 34 inequality constrained problems in [1], as well as a number of larger engineering applications. One typical application is described in [9]. The largest engineering application of the algorithm to date occurred in an optimum solid rocket motor design problem which involved 48 variables and 83 constraints.

---

[8] J. T. Betts, "Algorithm (To Appear): The Stationary Point of a Quadratic Function Subject to Linear Constraints," ACM Trans. Math. Software.

[9] J. T. Betts, "Optimal Three Burn Orbit Transfer," AIAA Journal, Vol 15, No. 6 (June 1977).

5. SUMMARY

An algorithm for computing the stationary point of a quadratic function of n variables subject to m linear equality constraints is developed. The algorithm has been implemented in FORTRAN. The implementation is compact since it requires no two-dimensional arrays beyond that needed to define the problem. The algorithm avoids mathematical operations which would degrade the conditioning of the original problem by utilizing an orthogonal decomposition of the constraint matrix. The solution generated by the algorithm is characterized by three properties: (a) the constraints are satisfied, (b) the norm of the gradient of the objective function restricted to the constraint surface is minimized and, (c) among all solutions satisfying the first two properties, the minimum length solution is chosen. When the stated problem has a unique solution, satisfaction of the first two properties defines the point. Nevertheless, the algorithm is not restricted to problems with definite Hessian matrices. The algorithm has been successfully tested as part of a general nonlinear programming algorithm.

APPENDIX

THE STATIONARY POINT OF A QUADRATIC FUNCTION SUBJECT TO LINEAR CONSTRAINTS

This algorithm implements the method developed in the preceding sections of this report.

```
      SUBROUTINE HSQP(A,B,C,D,M,N,TAU,G,H,U,IP,MAXRA,MAXRC,DJNORM,X,
     $   KRANK)
C
C
C
      DIMENSION B(1),D(1),G(1),H(1),U(1),IP(1),DJNORM(1),X(1)
      DIMENSION A(MAXRA,1),C(MAXRC,1)
C        PROGRAMMER AND DATE:  J.T.BETTS, JAN. 1978.
C
C        PURPOSE:  GIVEN AN M X N MATRIX C (OF RANK M), AN M VECTOR D,
C        AN N X N SYMMETRIC MATRIX A, AND AN N VECTOR B, FIND THE
C        STATIONARY POINT X OF THE QUADRATIC
C
C        J = .5*(X**T)*A*X + (B**T)*X
C
C        SUBJECT TO THE CONSTRAINTS
C
C        C*X = D.
C
C        IF A STATIONARY POINT DOES NOT EXIST THE ALGORITHM WILL FIND
C        A POINT WHICH SATISFIES THE CONSTRAINTS AND MINIMIZES THE
C        NORM OF THE GRADIENT OF J PROJECTED ON THE CONSTRAINT SURFACE.
C
C        ALGORITHM:  ORTHOGONAL DECOMPOSITION OF C MATRIX USING
C        HOUSEHOLDER TRANSFORMATIONS, FOLLOWED BY APPLICATION OF THE
```

```
C          OPTIMALITY CONDITIONS IN THE REDUCED VARIABLES.
C
C          INPUT:
C
C              A       N X N SYMMETRIC HESSIAN MATRIX
C              B       N DIMENSIONAL GRADIENT VECTOR
C              C       M X N JACOBIAN MATRIX (RANK M)
C              D       M DIMENSIONAL CONSTRAINT VECTOR
C              M       THE NUMBER OF CONSTRAINTS
C              N       THE NUMBER OF VARIABLES
C              TAU     PSEUDORANK TEST PARAMETER.  FOR A MACHINE WITH K
C                      SIGNIFICANT FIGURES AN APPROPRIATE VALUE IS
C                      TAU = 1.E-(K+2).
C              G       AUXILLIARY STORAGE (LENGTH M)
C              H       AUXILLIARY STORAGE (LENGTH N-M)
C              U       AUXILLIARY STORAGE (LENGTH N-M)
C              IP      AUXILLIARY STORAGE (LENGTH N-M)
C              MAXRA   MAXIMUM ROW DIMENSION OF A (MAXRA   N)
C              MAXRC   MAXIMUM ROW DIMENSION OF C (MAXRC   M)
C
C          OUTPUT:
C
C              DJNORM  PROJECTED GRADIENT NORM (ZERO IF X IS A STATIONARY
C                      POINT, NEGATIVE IF THERE IS AN INPUT ERROR)
C              X       COMPUTED STATIONARY POINT
C              KRANK   PSEUDORANK OF PROJECTED HESSIAN MATRIX (K2**T)*A*K2.
C                      WHEN KRANK .LT. N-M THE PROJECTION OF X ON THE
C                      CONSTRAINT SURFACE HAS MINIMUM NORM.
C
C          NOTE:       THE INPUT VALUES OF A,B,C, AND D ARE DESTROYED.
C
C ----------------------------------------------------------------------
C
C          INITIALIZATION
C
      KRANK = 0
      MP1 = M + 1
      NMM = N - M
      DJNORM(1) = -1.
C
C          CHECK FOR INPUT ERRORS
C
      IF(N.EQ.0.OR.N.GT.MAXRA.OR.M.GT.MAXRC.OR.M.GT.N) RETURN
C
```

```
C         IF THE PROBLEM IS UNCONSTRAINED GO TO STEP 7
C
      IF(M.EQ.0) GO TO 100
C
C ---------------------------------------------------------------------
C
C         STEP. 1.  COMPUTE ORTHOGONAL MATRIX K.  TRIANGULARIZE C.
C
      DO 10 I = 1,M
      CALL H12(1,I,I+1,N,C(I,1),MAXRC,G(I),C(I+1,1),MAXRC,1,M-I)
 10   CONTINUE
      IF(M.EQ.N) GO TO 50
C
C ---------------------------------------------------------------------
C
C         STEP 2.  COMPUTE ATILDA = (K**T)*A
C
      DO 20 I = 1,M
      CALL H12(2,I,I+1,N,C(I,1),MAXRC,G(I),A,1,MAXRA,N)
 20   CONTINUE
C
C ---------------------------------------------------------------------
C
C         STEP 3.  FORM THE LAST N-M ROWS OF AHAT = ATILDA*K;  I.E.
C         COMPUTE A21HAT = (K2**T)*A*K1 AND A22HAT = (K2**T)*A*K2
C
      DO 30 I = 1,M
      CALL H12(2,I,I+1,N,C(I,1),MAXRC,G(I),A(MP1,1),MAXRA,1,NMM)
 30   CONTINUE
C
C ---------------------------------------------------------------------
C
C         STEP 4.  COMPUTE BTILDA = (K**T)*B
C
      DO 40 I = 1,M
      CALL H12(2,I,I+1,N,C(I,1),MAXRC,G(I),B,1,1,1)
 40   CONTINUE
C
C ---------------------------------------------------------------------
C
C         STEP 5.  COMPUTE Y1HAT BY SOIVING THE LOWER TRIANGULAR
C         SYSTEM C*Y1 = D.  STORE IN X.
C
 50   CONTINUE
```

```
      X(1) = D(1)/C(1,1)
      IF(M.EQ.1) GO TO 80
      DO 70 I = 2,M
      IM1 = I - 1
      X(I) = D(I)
      DO 60 J = 1,IM1
      X(I) = X(I) - C(I,J)*X(J)
 60   CONTINUE
      X(I) = X(I)/C(I,I)
 70   CONTINUE
 80   CONTINUE
C
C         WHEN THERE ARE NO DEGREES OF FREEDOM GO TO STEP 8
C
      IF(M.EQ.N) GO TO 140
C
C -----------------------------------------------------------------------
C
C         STEP 6.  COMPUTE B2HAT = -B2TILDA - A21HAT*Y1HAT
C
      DO 90 I = MP1,N
      B(I) = -B(I)
      DO 90 J = 1,M
      B(I) = B(I) - A(I,J)*X(J)
 90   CONTINUE
C
C -----------------------------------------------------------------------
C
C         STEP 7. SOLVE A22HAT*Y2 = B2HAT FOR Y2 USING HFTI
C
 100  CONTINUE
C
C         COMPUTE PSEUDORANK TEST PARAMETER EPS
C
      EPS = TAU
      DO 120 J = MP1,N
      COLNRM = 0.
      DO 110 I = MP1,N
      COLNRM = COLNRM + A(I,J)**2
 110  CONTINUE
      EPS = AMAX1(EPS,TAU*SQRT(COLNRM))
 120  CONTINUE
C
      CALL HFTI(A(MP1,MP1),MAXRA,NMM,NMM,B(MP1),1,1,EPS,KRANK,DJNORM,
```

```
     $      H,U,IP)
C
      DO 130 I = MP1,N
      X(I) = B(I)
  130 CONTINUE
C
C         IF THE PROBLEM IS UNCONSTRAINED, RETURN.
C
      IF(M.EQ.0) RETURN
C
C -----------------------------------------------------------------------
C
C         STEP 8.  COMPUTE X = K*Y
C
  140 CONTINUE
      DO 150 K = 1,M
      I = MP1 - K
      CALL H12(2,I,I+1,N,C(I,1),MAXRC,G(I),X,1,1,1)
  150 CONTINUE
C
      RETURN
      END
```